**Problem-1.** Certain web pages are loading slow in user’s browser for our live web application. What steps will you take to resolve the issue?

**Answer:** I will take below steps to resolve the slow page loading issue in user’s browser.

1. Code Review: I make sure my developer teams are using all the tools at their disposal-from automated tools like profilers to best programming practices like code reviews. Poorly written code can lead to a host of web application issues including inefficient algorithms, memory leaks and application deadlocks. Old versions of software, or integrated legacy systems can also drag performance down.
2. Databases Optimization: I will make sure to use scripts and file statistics to check for any inefficient queries. Un-optimized database brings can destroy a production application. An optimized database allows for the highest levels of security and performance, while an Missing indexes slow down the performance of SQL queries causing, which can drag down an entire site.
3. Performance tuning: Systems must be properly tuned. Every setting should be checked: review thread counts, allocated memory and permissions. Confirm that all configuration parameters suit the demands placed on your web application, and aren’t the way they are just out of convenience. While default configurations make it easy to get new components up and running, they’re not always appropriate for your web applications in a live production environment.
4. Review DNS, Firewall, and Network Connectivity: DNS queries make up the majority of web traffic. That’s why a DNS issue can cause so much trouble, preventing visitors from accessing site and resulting in errors, 404s and incorrect pathways. Likewise, network connectivity and firewall efficiency are crucial for access and productivity. Using DNS monitoring safeguards to pinpoint problems at hand. Also, revise switches, check VLAN tags, and distribute tasks between servers. These are just a few ways to troubleshoot these types of performance issues.
5. Review Shared Resources and Virtual Machines: Just about every web application today relies on virtual machines for everything from scalability to management to system recovery. However, sometimes the way these virtual systems are organized – hundreds of VMs on a single physical server – can result in problems where one bogged-down system affects all the others. After all, contention is bound to happen. Monitor systems closely so that if one VM is causing problems, you can deal with the side-effects quickly.

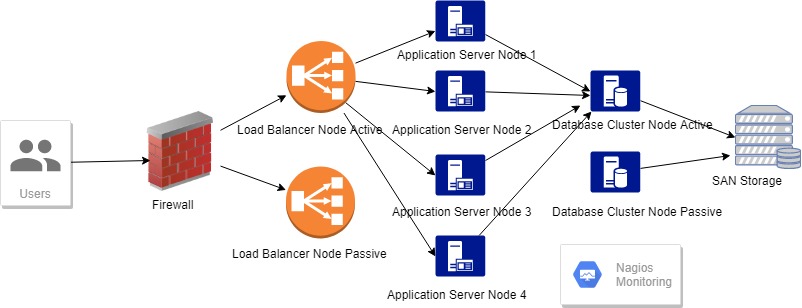
**Problem-2.** Imagine a scenario where a web application is serving from a single web server to the internet. What are the problems in this scenario? Design and architect a solution that will mitigate these problems? Or How would you design a scalable architecture with resiliency in mind for the following situations:

a. if a service is resource intensive b. a service needs to be low latency

c. if parts of a service need to be restricted to certain geographical boundaries

**Answer:** Considering the requirement of the scenario I will design a horizontally scalable architecture with high-availability (HA). Properly designing HA web applications is a complex task due to the overwhelming number of components and failure scenarios that can arise. In the real world, there is a large variance between deployments because virtually every web application has its own set of requirements. Based on the the most common components and also some best practices when designing a web server architecture for a high-availability site. It will also highlight elements that should consider when designing such architectures. An architect should carefully examine each component of their application and ensure that it is properly load balanced, fault tolerant, and scalable. The most basic architecture that will provide a high-availability is the following 3-tier setup. It is a load-balanced 6-server setup with fault tolerance and database backups that are stored SAN storage.

Below is the architecture diagram its beefing is under.



**Firewall:**

On the Firewall basically I will configure one to one NAT from the public IP to a private IP to use that as VIP to Load Balancer. And this firewall will restricted to certain geographical boundaries and other firewall functionalities.

**Front Ends / Load Balancers:**

Depending on the complexity of yousite, you could either have FrontEnd servers that act as your load balancers and application servers or a more complex setup where the amount of incoming traffic warrants the use of two dedicated load balancers, you could either have FrontEnd servers that act as your load balancers and application servers or a more complex setup where the amount of incoming traffic warrants the use of two dedicated load balancers.

Load balancers (LB) provide many advantages to serving client requests directly from your application servers, because they allow you to better control how traffic is handled in order to provide optimal performance. Most importantly, load balancers distribute requests between application servers and monitor them to ensure that client requests are always directed to a healthy machine.

I found HAProxy to be the highest performance, most reliable and dependable software load balancer, especially when you need to scale very quickly. When HAProxy is used in conjunction with Apache, it allows you to handle SSL connections and filter incoming requests to serve static content locally or direct traffic to one or more server arrays. This additional use of Apache adds a lot of flexibility with only a negligible amount of latency. Our default load balancers are configured to run in this manner. By default, clients round-robin through the load balancers by a DNS A-name record. In the unlikely event that a load balancer goes down, clients will simply retry the next address returned after a browser-specific timeout. I can also configure a hot-swappable standby, which monitors your load balancer's availability and I can configure URL based load balancing.

**Application Servers:**

In order to design a robust application, you need to ensure that it will be able to gracefully handle machine failure. What happens if one of the application servers is unresponsive?

If your application does not require session persistence, then this is not a concern, since the load balancer will be able to detect this condition and remove the faulty server from the rotation, ensuring that user requests are always sent to a healthy machine.

However, if your application does require session persistence, then you will need to properly address this issue. Generally, the best solution is to store session data in the database. This allows clients to transparently move between application servers while maintaining their session state. HAProxy is able to tie a user to a specific server throughout the duration of their session. While this may be good enough for some applications, it can cause clients to lose their session if an application server becomes no longer available due to accidental termination or an automatic scale-down of your application server array.

**Database Servers:**

For the majority of web applications, it is absolutely essential to run a reliable database. DB Clustering provides the necessary master/slave Server that are already preconfigured for automatic replication and regular backups. They are also designed to easily handle a variety of failover scenarios, such as a failure of the Master-DB. If your application is read intensive, then the best way to scale is to launch multiple slaves and have your application round-robin its reads through these slaves using a DNS A-name record. If you have a static number of slaves, this can be done with the default templates. However, if you want your slaves to scale, you will need to create a script that will register and deregister A records when your servers launch and when they are terminated.

If you are running a non-relational database, such as MySQL cluster, then you can simply add more nodes into your deployment. If you are running a relational database then usually the best way to scale your database is to run multiple master/slave pairs. This is typically the only way to increase the amount of writes that your deployment can handle. This requires partitioning your database and having your application decide which master to direct requests to.

**SAN Storage:**

In general, it's a good idea to leave plenty of storage capacity available for growth of the tablespaces. Even so, there will come a time when a database tablespace runs out of storage capacity and you will need to add more space. This is a situation where you may be better off using larger storage subsystems that supports many LUNs. With larger subsystems, it's easy to assign additional capacity simply by addressing the new LUN through an existing connection in the SAN. If you don't have the ability to add a LUN in an existing subsystem port, then you'll need to use additional switch and subsystem ports. For this reason, it's always a good idea to have capacity expansion ports reserved in your switches.

**Nagios Server for Minitoring:**

Nagios provides monitoring of all mission-critical infrastructure components including applications, services, operating systems, network protocols, systems metrics, and network infrastructure. Hundreds of third-party addons provide for monitoring of virtually all in-house applications, services, and systems.

**Problem-3**. Currently there’s no monitoring in place for the above single web server. How and what application will you use to monitor the resources/process in your new design?

**Answer:** I will prefer to commission a Nagios Server to monitor all above servers resource and processes. The powerful Nagios monitoring engine provides users with the highest degree of monitoring server performance resource and process. High-efficiency worker processes allow for nearly limitless scalability and monitoring effectiveness.Automated, integrated trending and capacity planning graphs allow organizations to plan for infrastructure upgrades before outdated systems catch them by surprise. Alerts are sent to IT staff, business stakeholders, and end-users via email or mobile text messages, providing them with outage details so they can start resolving issues immediately.

**Problem-4**. In our server we want to create a user who can only view logs using `less` from this path

/var/log. Please explain how to achieve this.

**Answer:** I will create a user with shell permission. And will give the user only read permission on /var/log/\* to acive this the command will be like

#useradd logreader

#passwd logreader

Password:

Confirm Password:

# setfacl -m u:logreader:r-- /var/log/\*

Now the user can read all under the file path /var/log

**Problem-5**. Explain how you can ssh into a private server from the internet.

Answer: I will use tunneling or port from router to ssh the private server from internet.

**Problem-6**. Write a bash function that will find all occurrences of an IPv4 from a given file.

Answer: Using grep command will find all occurrences of an IPv4 from a given file.

The command is below

grep -E -o "([0-9]{1,3}[\.]){3}[0-9]{1,3}" givenfilename.txt